RECEIVED
CENTRAL FAX CENTER
No. 0710 P. 1
FEB 10 2006

HAYNES BEFFEL & WOLFELD LLP
INTELLECTUAL PROPERTY LAW
P.O. Box 366, 751 Kelly Street
Half Moon Bay, CA. 94019
Phone: 650-712-0340 Fax 650-712-0263
www.HMBay.com

MARK HAYNES
ERNIE BEFFEL
WARREN WOLFELD
JIM HANN

Of Counsel:
*BILL KENNEDY
KENTA SUZUE
PETER SU
**JOSEPH E. ROOT

*admitted in MA only
**admitted in NY only

FACSIMILE TRANSMITTAL FORM

FAX NUMBER:	(571) 273-8300
TRANSMITTED TO:	Mail Stop Appeal Brief - Patents
OF:	United States Patent and Trademark Office
FROM:	Kenta Suzue Reg. No. 45,145
CLIENT/MATTER:	OUR FILE: 3COM 3715-1
DATE:	10 February 2006

TOTAL NUMBER OF PAGES INCLUDING THIS COVER SHEET: 24

ORIGINAL:	
X	WILL NOT FOLLOW
	FOLLOWS VIA AIRMAIL
	FOLLOWS VIA COURIER

Application No. 09/916,715 Atty Docket No.: 3COM 3715-1

CERTIFICATE OF FACSIMILE TRANSMISSION
I hereby certify that this correspondence is being facsimile transmitted to the United States Patent and Trademark Office at Facsimile No. (571) 273-8300 on 10 February 2006.

Amy Jonsson

IN THE UNITED STATES PATENT AND TRADEMARK OFFICE

In re Application of: Chi-Lie Wang et al.	Group Art Unit: 2663
Application No. 09/916,715	Examiner: Richard Chang
Confirmation No. 8232	**CUSTOMER NO. 22470**
Filed: 27 July 2001	
Title: **Network Interface Supporting of Virtual Paths for Quality of Service with Dynamic Buffer Allocation**	

MAIL STOP APPEAL BRIEF - PATENTS
Commissioner for Patents
P.O. Box 1450
Alexandria, VA 22313-1450

APPEAL BRIEF

Sir:

This Appeal Brief is filed in support of Appellants' appeal from the Final Office Action, mailed 10 August 2005 in this case. A Notice of Appeal was mailed on 10 November 2005.

The appropriate fee as set forth in 41.20(b)(2) of $500.00 and an extension of time as set forth in 1.136(a)(1) of $120.00 are covered in the attached Form PTO-2038 (Credit Card Payment Form).

Should it be determined that any fees are required in connection with this communication, the Commissioner is hereby authorized to charge those fees to Deposit Account No. 50-0869 (Attorney Docket No. 3COM 3715-1).

02/13/2006 MBINAS 00000027 09916715
01 FC:1402 500.00 OP

TABLE OF CONTENTS

I. REAL PARTY IN INTEREST

The real party in interest is 3Com Corporation, the assignee of record.

II. RELATED APPEALS AND INTERFERENCES

There are no known appeals or interferences relating to this case.

III. STATUS OF CLAIMS

Claims 1-48 are pending in this case. Claims 1, 6-17, 22-33, and 38-48 are rejected. Claims 2-5, 18-21, and 34-37 are objected to as depending upon a rejected base claim. Appellant is appealing the rejection of claims 1, 6-17, 22-33, and 38-48.

IV. STATUS OF AMENDMENTS

No amendments have been filed subsequent to the Final Office Action.

V. SUMMARY OF CLAIMED SUBJECT MATTER

The present application claims technology related to a network interface between a host processor and a network. Application at paragraph 3. Packets that traverse the network are associated with different quality of service levels. In one embodiment, there are three different quality of service levels: real time traffic, normal traffic, and IPSec traffic. Application at paragraph 49. Real time traffic requires more immediate attention than other types of traffic. Application at paragraph 38. Packets waiting to be transmitted from the network interface out of a single physical port can be divided into multiple virtual paths, depending on the quality of service of the packets. Id. Higher priority traffic in a high priority virtual path can be transmitted out the single physical port prior to lower priority traffic in lower priority virtual paths. Id. One key part of the network interface is the sizes of the virtual paths. If a memory buffer supporting a particular virtual path is overused, the buffer may be overrun, blocking incoming traffic. Application at paragraph 86. If a memory buffer supporting a particular virtual path is underused, the buffer resources are wasted. Id. The invention claimed in the present application addresses the sizes of the virtual paths in a manner that does not require the virtual paths to remain at a constant size.

Claims 1, 17, and 33 are the independent claims of the present application, and are summarized below.

Claim 1

Claim 1 describes a computer system coupled to a network. The computer system includes a host processor and a network interface coupled to the network and to the host processor. The network interface includes first and second ports that, respectively, receive data from the host processor and transmit data to the network. The network interface also includes a memory, which is coupled to the first and second ports, that stores data packets received by the first port. A control circuit on the network interface manages the memory as queues having respective priorities. Logic on the control circuit places a packet received from the host into one of the queues according to a quality of service parameter associated with the packet, and services packets in the queues according to the respective priorities.

The network interface of claim 1 also includes logic that dynamically allocates space in the memory to the queues. Because the queues of respective priorities have dynamically allocated memory space, the memory that was previously allocated to a less busy queue can be shifted to a busier queue. In this fashion, the memory previously allocated to a less busy queue need not remain underused, and the busier queue is less likely to be filled and overrun.

Claim 17

Claim 17 recites the invention in the form of a method for managing the transfer of data packets between a host processor and a network. The two steps of the method are similar to the language of claim 1 describing the network interface circuitry of claim 1. First, managing memory in the network interface apparatus as queues having respective priorities, including placing a packet received from the host processor into one of the queues according to a quality of service parameter associated with the packet, and servicing packets in the queues according to the respective priorities. Second, dynamically allocating space in the memory to the queues.

Claim 33

Claim 33 recites an integrated circuit for use in a network interface between a host processor and a network. The recitation of claim 33 is like that of claim 1, but does not require the host processor.

VI. GROUNDS OF REJECTION TO BE REVIEWED ON APPEAL

Whether claims 1, 6-17, 22-33, and 38-48 are unpatentable under 35 U.S.C. 102(e) as being anticipated by Yavatkar et al. (U.S. Patent No. 6,728,265, hereinafter "Yavatkar").

VII. ARGUMENT

Claims 1, 17, and 33

Claim 1 includes at least the following limitation not found in Yavatkar:

> a memory that stores data packets received by the first port, the memory being coupled to the first port and to the second port;
>
> ....
>
> logic to dynamically allocate space in said memory to the queues in the plurality of queues.

Yavatkar discloses a network controller 52 which has a transmit path 92 including one or more FIFO memories 122 for packets in the transmit path. column 6, lines 14-16. Yavatkar also discusses multiple queues associated with different priority levels. The particular queue from which the next transmission occurs depends on factors such as the priority of the multiple queues. column 6, lines 31-50. The statement that there are one or more FIFO memories suggests that each priority level queue has its own physical FIFO memory. Yavatkar includes a software mechanism that modifies existing packets in a queue, to "expedite the transmission of an urgent packet ahead of other packets". column 7, lines 1-18 and Fig. 5B. There is no discussion in Yavatkar which suggests that the software mechanism changes the sizes of the queues.

The Examiner takes the mistaken position that the software mechanism described in column 7, lines 1-18 and Fig. 5B corresponds with the "logic to dynamically allocate space in said memory to the queues" recited in claim 1. The software mechanism does not change the space in memory allocated to the queues, such that the queues hold fewer packets or more packets. Rather than changing the queue itself, the software mechanism modifies packets that already exist in the queue.

The Examiner also takes the mistaken position that the FIFO memories 122 correspond with the "memory" recited in claim 1, which also recites "logic" that

dynamically allocates space in the "memory" to the queues. The statement in Yavatkar that that there are one or more FIFO memories suggests that each priority level queue has its own physical FIFO memory. Accordingly, rather than having space in the FIFO memories dynamically allocated to the queues, each priority level queue in Yavatkar is statically and permanently allocated the size of its associated FIFO memory.

Accordingly, Yavatkar does not anticipate claim1, and does not anticipate claims 6-16 which depend from claim 1.

Independent claim 17 also requires dynamic allocation of space to queues, and a memory whose space is dynamically allocated. The Examiner did not provide bases separate from those applied to claim 1, in support of the rejection of claim 17. Applicant submits that claim 17 is patentable for at least the reasons discussed above with respect to claim 1. Likewise, claims 22-32 which depend from claim 17 are not anticipated by Yavatkar.

Independent claim 33 also requires logic that dynamically allocates space to queues, and a memory whose space is dynamically allocated. The Examiner did not provide bases separate from those applied to claim 1, in support of the rejection of claim 33. Applicant submits that claim 33 is patentable for at least the reasons discussed above with respect to claim 1. Likewise, claims 38-48 which depend from claim 33 are not anticipated by Yavatkar.

<u>Claims 9, 25, and 41</u>

Claim 9 includes at least the following limitation not found in Yavatkar:

> wherein said logic ... maintains a list of free buffers and a list of used buffers for each of the plurality of queues

Claim 9 depends from claim 1, and therefore includes the limitations of claim 1. Claim 1, and therefore claim 9, includes the following limitation not found in Yavatkar:

> a control circuit that manages the memory as a plurality of queues having respective priorities, including logic to place a packet received <u>from the host</u> into one of the plurality of queues according to a quality of service parameter associated with the packet, and logic to service packets in the plurality of queues according to the respective priorities (emphasis added)

As shown above, the queues in claim 9 receive packets from the host.

The Examiner's argument is that the driver program 57 dynamically allocates space in a FIFO memory and inherently maintains a list of used buffers and free buffers for the queues. The Examiner's argument is mistaken, because the driver program 57 in Yavatkar is associated with flow tuples of data sent to the host. column 3, lines 59-62. As shown by Fig. 5, the flow tuples in Yavatkar only relate to the receive path 92 for data sent to the host connected to PCI bus 72, and not to the transmit path 94 for data sent from the host connected to the PCI bus 72. In contrast with Yavatkar, the queues in claim 9 receive packets "from the host". It is also unclear why the Examiner believes that the driver program 57 inherently corresponds to maintaining "a list of free buffers and a list of used buffers" as claimed in claim 9.

The Examiner did not provide bases separate from those applied to claim 9, in support of the rejection of claims 25 and 41. Applicant submits that claims 25 and 41 are patentable for at least the reasons discussed above with respect to claim 9.

Claims 10-11, 26-27, and 42-43

Regarding claims 10-11, 26-27, and 42-43, the Examiner alleged that the limitations are similar to those of claims 9, 25, and 41, and rejected claims 10-11, 26-27, and 42-43 with the same rationale applied against claims 9, 25, and 41. The Examiner is incorrect, because the limitations of claims 10-11, 26-27, and 42-43 are distinct from those of claims 9, 25, and 41. Accordingly, the Examiner failed to show that Yavatkar teaches every element of claims 10-11, 26-27, and 42-43. Applicant submits that claims 10-11, 26-27, and 42-43 are patentable.

Claims 12-13, 28-29, and 44-45

Regarding claims 12-13, 28-29, and 44-45, the Examiner alleged that the limitations are similar to those of claims 9, 25, and 41, and rejected claims 12-13, 28-29, and 44-45 with the same rationale applied against claims 9, 25, and 41. The Examiner is incorrect, because the limitations of claims 12-13, 28-29, and 44-45 are distinct from those of claims 9, 25, and 41. Accordingly, the Examiner failed to show that Yavatkar teaches every element of claims 12-13, 28-29, and 44-45. Applicant submits that claims 12-13, 28-29, and 44-45 are patentable.

Claims 14-15, 30-31, and 46-47

Regarding claims 14-15, 30-31, and 46-47, the Examiner alleged that the limitations are similar to those of claims 9, 25, and 41, and rejected claims 14-15, 30-31, and 46-47 with the same rationale applied against claims 9, 25, and 41. The Examiner is incorrect, because the limitations of claims 14-15, 30-31, and 46-47 are distinct from those of claims 9, 25, and 41. Accordingly, the Examiner failed to show that Yavatkar teaches every element of claims 14-15, 30-31, and 46-47. Applicant submits that claims 14-15, 30-31, and 46-47 are patentable.

Claims 16, 32, and 48

Regarding claims 16, 32, and 48, the Examiner alleged that the limitations are similar to those of claims 9, 25, and 41, and rejected claims 16, 32, and 48 with the same rationale applied against claims 9, 25, and 41. The Examiner is incorrect, because the limitations of claims 16, 32, and 48 are distinct from those of claims 9, 25, and 41. Accordingly, the Examiner failed to show that Yavatkar teaches every element of claims 16, 32, and 48. Applicant submits that claims 16, 32, and 48 are patentable.

CONCLUSION

In view of the foregoing, Appellants ask that this honorable Board reverse the Examiner's rejections of the claims. In addition, it is submitted that all claims which are the subject of this examination are now allowable, and a notice of intent to issue a patent is respectfully requested.

The Commissioner is hereby authorized to charge any fee determined to be due in connection with this communication, or credit any overpayment, to our Deposit Account No. 50-0869 (File No. 3COM 3715-1).

Respectfully submitted,

Dated: 10 February 2006

Kenta Suzue, Reg. No. 45,145
Attorney for Patent Owner

HAYNES BEFFEL & WOLFELD LLP
P.O. Box 366
751 Kelly Street
Half Moon Bay, CA 94019
Telephone: (650)712.0340
Facsimile: (650)712.0263

CLAIMS APPENDIX

1. (Rejected) A computer system, comprising:

a host processor, and

a network interface coupled to the host processor and to a network, the network interface comprising:

a first port that receives data from the host processor;

a second port that transmits data to the network;

a memory that stores data packets received by the first port, the memory being coupled to the first port and to the second port;

a control circuit that manages the memory as a plurality of queues having respective priorities, including logic to place a packet received from the host into one of the plurality of queues according to a quality of service parameter associated with the packet, and logic to service packets in the plurality of queues according to the respective priorities; and

logic to dynamically allocate space in said memory to the queues in the plurality of queues.

2. (Objected to) The computer system of claim 1, wherein the plurality of queues includes a higher priority queue, and a lower priority queue, and including a timeout timer coupled with the lower priority queue which is enabled if a packet is stored in the lower priority queue and expires after a timeout interval, and including logic to preempt the higher priority queue in favor of the lower priority queue if the timeout timer expires.

3. (Objected to) The computer system of claim 1, wherein the plurality of queues includes a higher priority queue, an intermediate priority queue, and a lower priority queue, and including

a first timeout timer coupled with the intermediate priority queue which is enabled if a packet is stored in the intermediate priority queue and expires after a first timeout interval, and including logic to preempt the higher priority queue in favor of the intermediate priority queue if the first timeout timer expires; and

a second timeout timer coupled with the lower priority queue which is enabled if a packet is stored in the lower priority queue and expires after a second timeout interval, and including logic to preempt the higher priority queue and the intermediate priority queue in favor of the lower priority queue if the second timeout timer expires.

4. (Objected to) The computer system of claim 1, wherein the plurality of queues includes a higher priority queue, an intermediate priority queue, and a lower priority queue, and including

an first timeout timer coupled with the intermediate priority queue which is enabled if a packet is stored in the intermediate priority queue and expires after a first timeout interval, and including logic to preempt the higher priority queue in favor of the intermediate priority queue if the first timeout timer expires;

a second timeout timer coupled with the lower priority queue which is enabled if a packet is stored in the lower priority queue and expires after a second timeout interval, and including logic to preempt the higher priority queue and the intermediate priority queue in favor of the lower priority queue if the second timeout timer expires; and

logic to service the intermediate priority queue in favor of the lower priority queue if both the first and second timeout timers expire.

5. (Objected to) The computer system of claim 1, further comprising logic in the network interface to execute a security process on packets in one of the plurality of queues.

6. (Rejected) The computer system of claim 1, wherein the second port further comprises circuitry for formatting packets according to a protocol compliant with an Ethernet protocol standard.

7. (Rejected) The computer system of claim 1, wherein the second port further comprises circuitry for formatting packets according to a protocol compliant with an InfiniBand protocol standard.

8. (Rejected) The computer system of claim 1, wherein the packets include frame start headers, and said quality of service parameters comprises codes in the frame start headers.

9. (Rejected) The computer system of claim 1, wherein said logic to dynamically allocate space in said memory maintains a list of free buffers and a list of used buffers for each of the plurality of queues.

10. (Rejected) The computer system of claim 1, wherein said memory comprises a single storage array.

11. (Rejected) The computer system of claim 1, wherein said logic to dynamically allocate space in said memory, places packets downloaded to the memory in a plurality of buffers in non-contiguous memory locations.

12. (Rejected) The computer system of claim 1, wherein said logic to dynamically allocate space in said memory maintains a list of free buffers and a list of used buffers for each of the plurality of queues, so that each virtual path has a free buffer list having a number of free buffers, and includes logic which releases a used buffer to the free buffer list for a queue in the plurality of queues having a smallest number of free buffers.

13. (Rejected) The computer system of claim 1, wherein said logic to dynamically allocate space in said memory maintains a list of free buffers and a list of used buffers for each of the plurality of queues, so that each virtual path has a free buffer list having a number of free buffers, and includes logic which releases a used buffer to the free buffer list for a queue in the plurality of queues having a largest amount of traffic.

14. (Rejected) The computer system of claim 1, wherein logic to dynamically allocate space in said memory maintains a list of buffer descriptors for corresponding buffers in said memory, the buffer descriptors including a parameter specifying a size of the corresponding buffer, and a location of the corresponding buffer.

15. (Rejected) The computer system of claim 1, wherein logic to dynamically allocate space in said memory maintains a list of buffer descriptors for corresponding buffers in said memory, the buffer descriptors including a programmable parameter specifying a size of the corresponding buffer.

16. (Rejected) The computer system of claim 1, wherein at least one queue in the plurality of queues comprises a first-in-first-out FIFO queue.

17. (Rejected) In a network interface apparatus, a method for managing transfer of data packets between a host processor and a network, comprising:

managing memory in the network interface apparatus as a plurality of queues having respective priorities, including placing a packet received from the host processor into one of the plurality of queues according to a quality of service parameter associated with the packet, and servicing packets in the plurality of queues according to the respective priorities; and

dynamically allocating space in said memory to the queues in the plurality of queues.

18. (Objected to) The method of claim 17, wherein the plurality of queues includes a higher priority queue, and a lower priority queue, and including enabling a timeout timer coupled with the lower priority queue if a packet is stored in the lower priority queue, the timeout timer expiring after a timeout interval, and preempting the higher priority queue in favor of the lower priority queue if the timeout timer expires.

19. (Objected to) The method of claim 17, wherein the plurality of queues includes a higher priority queue, an intermediate priority queue, and a lower priority queue, and including

enabling a first timeout timer coupled with the intermediate priority queue if a packet is stored in the intermediate priority queue, the first timeout timer expiring after a first timeout interval, and preempting the higher priority queue in favor of the intermediate priority queue if the first timeout timer expires; and

enabling a second timeout timer coupled with the lower priority queue if a packet is stored in the lower priority queue, the second timeout timer expiring after a second timeout interval, and preempting the higher priority queue and the intermediate priority queue in favor of the lower priority queue if the second timeout timer expires.

20. (Objected to) The method of claim 17, wherein the plurality of queues includes a higher priority queue, an intermediate priority queue, and a lower priority queue, and including

enabling a first timeout timer coupled with the intermediate priority queue if a packet is stored in the intermediate priority queue, the first timeout timer expiring after a first timeout interval, and preempting the higher priority queue in favor of the intermediate priority queue if the first timeout timer expires; and

enabling a second timeout timer coupled with the lower priority queue if a packet is stored in the lower priority queue, the second timeout timer expiring after a second timeout interval, and preempting the higher priority queue and the intermediate priority queue in favor of the lower priority queue if the second timeout timer expires; and

servicing the intermediate priority queue in favor of the lower priority queue if both the first and second timeout timers expire.

21. (Objected to) The method of claim 17, further comprising executing a security process on packets in one of the plurality of queues.

22. (Rejected) The method of claim 17, including formatting packets in the network interface device according to a protocol compliant with an Ethernet protocol standard.

23. (Rejected) The method of claim 17, including formatting packets in the network interface device packets according to a protocol compliant with an Infiniband protocol standard.

24. (Rejected) The method of claim 17, wherein the packets include frame start headers, and said quality of service parameters comprises codes in the frame start headers.

25. (Rejected) The method of claim 17, wherein said dynamically allocating space in said memory includes maintaining a list of free buffers and a list of used buffers for each of the plurality of queues.

26. (Rejected) The method of claim 17, wherein said memory comprises a single storage array.

27. (Rejected) The method of claim 17, wherein said dynamically allocating space in said memory includes placing packets downloaded to the memory in a plurality of buffers in non-contiguous memory locations.

28. (Rejected) The method of claim 17, wherein said dynamically allocating space in said memory includes maintaining a list of free buffers and a list of used buffers for each of the plurality of queues, so that each virtual path has a free buffer list having a number of free buffers, and includes releasing a used buffer to the free buffer list for a queue in the plurality of queues having a smallest number of free buffers.

29. (Rejected) The method of claim 17, wherein said dynamically allocating space in said memory includes maintaining a list of free buffers and a list of used buffers for each of the plurality of queues, so that each virtual path has a free buffer list having a number of free buffers, and includes releasing a used buffer to the free buffer list for a queue in the plurality of queues having a largest amount of traffic.

30. (Rejected) The method of claim 17, wherein said dynamically allocating space in said memory includes maintaining a list of buffer descriptors for corresponding buffers in said memory, the buffer descriptors including a parameter specifying a size of the corresponding buffer, and a location of the corresponding buffer.

31. (Rejected) The method of claim 17, wherein said dynamically allocating space in said memory includes maintaining a list of buffer descriptors for corresponding buffers

in said memory, the buffer descriptors including a programmable parameter specifying a size of the corresponding buffer.

32. (Rejected) The method of claim 17, including managing at least one queue in the plurality of queues as a first-in-first-out FIFO queue.

33. (Rejected) An integrated circuit for use in a network interface between a host processor and a network, comprising:

a first port that receives data from the host processor;

a second port that transmits data to the network;

a memory that stores data packets received by the first port, the memory being coupled to the first port and to the second port; and

a control circuit that manages the memory as a plurality of queues having respective priorities, including logic to place a packet received on the first port into one of the plurality of queues according to a quality of service parameter associated with the packet, and logic to transmit packets in the plurality of queues out the second port according to the respective priorities; and

logic to dynamically allocate space in said memory to the queues in the plurality of queues.

34. (Objected to) The integrated circuit of claim 33, wherein the plurality of queues includes a higher priority queue, and a lower priority queue, and including a timeout timer coupled with the lower priority queue which is enabled if a packet is stored in the lower priority queue and expires after a timeout interval, and including logic to preempt the higher priority queue in favor of the lower priority queue if the timeout timer expires.

35. (Objected to) The integrated circuit of claim 33, wherein the plurality of queues includes a higher priority queue, an intermediate priority queue, and a lower priority queue, and including

a first timeout timer coupled with the intermediate priority queue which is enabled if a packet is stored in the intermediate priority queue and expires after a first timeout

interval, and including logic to preempt the higher priority queue in favor of the intermediate priority queue if the first timeout timer expires; and

a second timeout timer coupled with the lower priority queue which is enabled if a packet is stored in the lower priority queue and expires after a second timeout interval, and including logic to preempt the higher priority queue and the intermediate priority queue in favor of the lower priority queue if the second timeout timer expires.

36. (Objected to) The integrated circuit of claim 33, wherein the plurality of queues includes a higher priority queue, an intermediate priority queue, and a lower priority queue, and including

an first timeout timer coupled with the intermediate priority queue which is enabled if a packet is stored in the intermediate priority queue and expires after a first timeout interval, and including logic to preempt the higher priority queue in favor of the intermediate priority queue if the first timeout timer expires;

a second timeout timer coupled with the lower priority queue which is enabled if a packet is stored in the lower priority queue and expires after a second timeout interval, and including logic to preempt the higher priority queue and the intermediate priority queue in favor of the lower priority queue if the second timeout timer expires; and

logic to service the intermediate priority queue in favor of the lower priority queue if both the first and second timeout timers expire.

37. (Objected to) The integrated circuit of claim 33, further comprising logic in the network interface to execute a security process on packets in one of the plurality of queues.

38. (Rejected) The integrated circuit of claim 33, wherein the second port further comprises media access control circuitry for transmitting packets according to a protocol compliant with an Ethernet protocol standard.

39. (Rejected) The integrated circuit of claim 33, wherein the second port further comprises media access control circuitry for transmitting packets according to a protocol compliant with an Infiniband protocol standard.

40. (Rejected) The integrated circuit of claim 33, wherein the packets include frame start headers, and said quality of service parameters comprises codes in the frame start headers.

41. (Rejected) The integrated circuit of claim 33, wherein said logic to dynamically allocate space in said memory maintains a list of free buffers and a list of used buffers for each of the plurality of queues.

42. (Rejected) The integrated circuit of claim 33, wherein said memory comprises a single storage array.

43. (Rejected) The integrated circuit of claim 33, wherein said logic to dynamically allocate space in said memory, places packets downloaded to the memory in a plurality of buffers in non-contiguous memory locations.

44. (Rejected) The integrated circuit of claim 33, wherein said logic to dynamically allocate space in said memory maintains a list of free buffers and a list of used buffers for each of the plurality of queues, so that each virtual path has a free buffer list having a number of free buffers, and includes logic which releases a used buffer to the free buffer list for a queue in the plurality of queues having a smallest number of free buffers.

45. (Rejected) The integrated circuit of claim 33, wherein said logic to dynamically allocate space in said memory maintains a list of free buffers and a list of used buffers for each of the plurality of queues, so that each virtual path has a free buffer list having a number of free buffers, and includes logic which releases a used buffer to the free buffer list for a queue in the plurality of queues having a largest amount of traffic.

46. (Rejected) The integrated circuit of claim 33, wherein logic to dynamically allocate space in said memory maintains a list of buffer descriptors for corresponding buffers in said memory, the buffer descriptors including a parameter specifying a size of the corresponding buffer, and a location of the corresponding buffer.

47. (Rejected) The integrated circuit of claim 33, wherein logic to dynamically allocate space in said memory maintains a list of buffer descriptors for corresponding buffers in said memory, the buffer descriptors including a programmable parameter specifying a size of the corresponding buffer.

48. (Rejected) The integrated circuit of claim 33, wherein at least one queue in the plurality of queues comprises a first-in-first-out FIFO queue.

Application No. 09/916,715 Atty Docket No.: 3COM 3715-1

EVIDENCE APPENDIX

None.

Application No. 09/916,715 Atty Docket No.: 3COM 3715-1

RELATED PROCEEDINGS APPENDIX

None.